**IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE**

PATENT APPLICATION

Applicant(s):	Kedar Sharadchandra Namjoshi
Case:	2
Serial No.:	10/614,618
Filing Date:	July 7, 2003
Group:	2193
Examiner:	Tuan A. Vu
Title:	Method and Apparatus for Reducing a Program Size While Maintaining Branching Time Properties and Automated Checking of Such Reduced Programs

---

REPLY BRIEF

Mail Stop Appeal Brief – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Appellant hereby replies to the Examiner's Answer, mailed May 28, 2008 (referred to hereinafter as "the Examiner's Answer"), in an Appeal of the final rejection of claims 1-26 in the above-identified patent application.

REAL PARTY IN INTEREST

A statement identifying the real party in interest is contained in Appellant's Appeal Brief.

RELATED APPEALS AND INTERFERENCES

A statement identifying related appeals is contained in Appellant's Appeal Brief.

STATUS OF CLAIMS

A statement identifying the status of the claims is contained in Appellant's Appeal Brief.

STATUS OF AMENDMENTS

A statement identifying the status of the amendments is contained in Appellant's Appeal Brief.

SUMMARY OF CLAIMED SUBJECT MATTER

A Summary of the Invention is contained in Appellant's Appeal Brief.

STATEMENT OF GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A statement identifying the grounds of rejection to be reviewed on appeal is contained in Appellant's Appeal Brief.

CLAIMS APPEALED

A copy of the appealed claims is contained in an Appendix of Appellant's Appeal Brief.

ARGUMENT

Appellant first notes that BenAri does *not* address the problem solved by the claimed invention. For example, BenAri teaches the following results:

(1) An exponential decision procedure for satisfiability;

(2) A finite model property: if a formula in UB is satisfiable then it has a finite (exponential) model;

(3) A simple axiomatization which is shown to be complete.

(See, bottom of page 208.)

Appellant notes that "satisfiability" asks whether there EXISTS a program which satisfies a formula. In one aspect of the present invention, on the other hand, a method is disclosed to check whether a GIVEN program M satisfies a formula, f. The method performs this check by constructing a reduced program, M', and an altered formula, f', such that the property is preserved, i.e., M satisfies property f if, and only if, M' satisfies property f'.

Appellant notes that the "finite model property" is a mathematical property of the logic UB, used in the satisfiability procedure. It states that, if there EXISTS a program satisfying a given formula, there EXISTS a finite-state program satisfying the same formula. This, as explained above, is about satisfiability checking, whereas, in the claimed invention, a SPECIFIC program is given.

Appellant also notes that "axiomatization" refers to a set of rules to determine if a formula is true over ALL programs. BenAri shows that rules A1-A4, E1-E4, R1-R3 on page 211 suffice to do so (Theorem 5.10, page 223).

Furthermore, Appellant notes that the claimed invention is based on a paper entitled "Abstraction for Branching Time Properties" accepted at the 15th International Conference on Computer Aided Verification (CAV 2003, Boulder, CO, USA, July 8-12, 2003). Appellant maintains that the acceptance of the paper at this premier conference is evidence that the claimed invention would *not* be obvious to a person of ordinary skill in the art.

Point A

The Examiner asserts that there are no further specifics as to how exactly the "with reduced number of states" have been achieved from the cited steps.

Appellant notes that the specification teaches how the reduced number of states is achieved. The cited claims simply require that the *abstract program is computed with a reduced number of states*. The current claim language is necessary to provide the Appellants the protection to which they are entitled.

Point B

The Examiner notes, essentially, that the taking of an automaton product is well known.

Appellant claims, for example, a method for reducing a program; Appellant does *not* claim to have discovered the taking of an automaton product.

Point C

The Examiner asserts that, since the claimed computing step is devoid of implementation details, the limitation as to "with a reduced number of states" appears to be an end result devoid of mechanics by which such end result has been attained. The Examiner asserts that, absent any details about 'altered' or 'reducing,' one would not be able to see how based on altering a version of a branching property f, a reduced number of states is accomplished.

Appellant notes that the limitation of "a reduced number of states" is a *characteristic of the abstract program* and *not* an end result. As noted above, the specification clearly teaches how the reduced number of states is achieved.

Points D and G

The Examiner's point is not easily understood, partly due to an error in the interpretation of BenAri. For example, the Examiner refers to "UB" as "the product process UxB." As BenAri teaches, however, "UB" is an abbreviation of the "Unified system of Branching time"; contrary to the Examiner's assertion, it is *not* a product process (see, bottom of page 208 of BenAri). In addition, the Examiner apparently asserts that a proof as disclosed in BenAri is a form of reduction. In the context of the claimed invention, this interpretation is *not* correct. The BenAri proof refers to proving that a property is true for ALL programs; in the context of the claimed invention, a property for A SPECIFIC program is proved. These two proofs are *not* equivalent, as would be apparent to a person of ordinary skill in the art.

Point E

The Examiner asserts that one of ordinary skill in the art would see the UB model as an analogue of the cross product of a program input and a branching time property.

Contrary to the Examiner's assertion, Appellant maintains that one of ordinary skill in the art would *not* recognize the *BenAri model as a product construction*.

Point F

The Examiner apparently asserts that the claimed invention does not show how a property is preserved.

Appellants note that the preservation of a property is an *attribute of the method.*

Point H

Similar to Point D above, the BenAri proof refers to proving that a property is true for ALL programs; in the context of the claimed invention, a property for A SPECIFIC program is proven. These two proofs are *not* equivalent, as would be apparent to a person of ordinary skill in the art.

Point I

The Examiner apparently equates an induction proof (or a proof system) which proves a property for ALL programs with a reduction that holds only for a SPECIFIC program.

Appellant notes, however, that an induction and reduction are *not* equivalent, as would be apparent to a person of ordinary skill in the art.

Appeal Brief Arguments

Specification Objection

The Examiner has requested hard copies of any NPL documents that may help the Examiner to better interpret and understand what is novel. Appellant again note that only non-essential material provided for background information has been incorporated by reference. Appellant would be more than happy to provide a hard copy of any document that would help the Examiner better understand the invention. It is difficult, however, for Appellant to ascertain which portions of the invention the Examiner is struggling with. As set forth in the above Summary section, for example, Appellant maintains that every claim limitation is fully supported and described within the specification.

Section 112 Rejection

Claims 1-26 were rejected under 35 U.S.C. §112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which

applicant regards as the invention. The Examiner asserts that the claims omit essential steps. The ***only*** basis for the Section 112 rejection seems to be the Examiner's contention that one of ordinary skill in the art would not be "able to learn (i) what is actually expressed as an automaton A; and (ii) whether the abstract program being computed is the result from combining reduced number of states and an altered version of one branch property f, OR this computed program yields reduced number of states and an altered version of a property."

Point (i)

Appellant submits that the specification and figures are entirely clear and address point (i). As clearly shown in step 210 of FIG. 2, and discussed in the specification on page 6, lines 19-22, the program abstraction method 200 is initiated with the program, M, to be abstracted, as well as the branching time property, f, to be preserved and an automaton, A, for f. Clearly, it is the branching time property, f, that is expressed as an automaton, A. Appellant submits that this is also entirely clear in claims 1 and 14.

Point (ii)

Both alternative interpretations suggested by the Examiner as part of point (ii) are inaccurate and inconsistent with the actual claim language in claims 1 and 14.

The Examiner and the Appeal Board are again referred to FIG. 2, where the program abstraction method 200 is initiated with the program, M, to be abstracted, as well as the branching time property, f, to be preserved. The branching time property, f, is expressed as an automaton, A.

The exemplary program abstraction method 200 provides two results, both recited in the third step (or limitation) of claims 1 and 14. First, the exemplary program abstraction method 200 computes an abstract program (referred to herein as Q) with a reduced number of states. In addition, the exemplary program abstraction method 200 computes an altered property (referred to herein as g). Thus, g is an "altered version of f," as recited in claim 1.

The Examiner's main source of confusion seems to be with the second step of claim 1. The examiner's alternatives interpretations are:

A. the abstract program being computed is the result from combining reduced number of states and an altered version of one branch property, f.

This interpretation is incorrect. Q is created from M and the *original* property, f.

B. this computed program yields reduced number of states and an altered version of a property.

This interpretation is incorrect in the sense that the computed program, Q, does not "yield" property g. Both Q and g are computed independently of one another from the given information.

The second step of claim 1 is described in conjunction with step 220 of FIG. 2, as well as page 6, lines 23-28, where it is described that:

> Thereafter, the user-specified abstract domain, $\overline{S}$, and a set of left-total abstraction relations $\{\xi_q | q \in Q\}$ are obtained during step 220, where each $\xi_q \subseteq S \times \overline{S}$. The abstract domain is a set of values to generalize the possible real values of the program. The abstraction relations relate the real program states to the abstract domain. ξ_n and ξ_{ff} are defined to be $S \times \overline{S}$, and note that (s, q) is related to (t, q) if and only if $s \xi_q t$ holds.

This passage clearly defines both the abstract domain and abstract relations, as recited in the second step of claim 1, in a completely consistent manner with the claim terminology.

Appellant respectfully requests withdrawal of the Section 112 rejection.

Prior Art Rejections: Independent Claims 1 and 14

Independent claims 1 and 14 were rejected under 35 U.S.C. §102(b) as being anticipated by BenAri. Regarding claim 1, the Examiner asserts that BenAri teaches a method for reducing a program, M, that preserves (citing page 5) at least one branching time property, f, comprising the steps of: forming a product of said program, M and said branching time property, f (citing middle of page 208), expressed as an automaton, A (citing bottom of page 208 and top

of page 209); obtaining an abstract domain containing a set of abstract values to generalize possible states of said program and abstract relations that relate said program states to said abstract domain (citing sec. 2, page 209, bottom, and Semantics for UB, page 210); and computing an abstract program with a reduced number of states (citing pages 209-224) and an altered version of said branching time property, f (citing pages 212-214), using said product (citing page 210, top, and page 211, top).

BenAri considers whether there exists a program for which a formula is true. BenAri take a formula of a branching time property and determines if it is true.

The present invention, on the other hand, given a program and a property, determines if the property is true of this program.

Among other important limitations found in claims 1 and 14, BenAri does not disclose or suggest *reducing* a program. It is noteworthy that the Examiner has not alleged that this is shown by **any** portion of BenAri.

In addition, BenAri does not disclose or suggest *preserving* a branch time property. Rather, BenAri is given a branch time property and identifies a program for which it holds.

BenAri also does not disclose or suggest forming a product of the program, M and said branching time property, f. It is again noteworthy that the Examiner has not alleged that **any** portion of BenAri describes a product.

BenAri additionally does not disclose an "*abstract domain* containing a set of abstract values to generalize possible states of said program." The Examiner is merely applying keyword spotting in this portion of the rejection.

BenAri also does not disclose or suggest "computing an abstract program with a reduced number of states." The Examiner has not stated how the formulas, axioms, theorems and lemmas read on the elimination of paths.

BenAri also does not disclose or suggest computing "an altered version of said branching time property, f." The Examiner has made no attempt to indicate how the branching time property referenced in the first step is altered to form the altered version referenced in the

final step.

Thus, other than matching a few key words from the claims of the present invention, BenAri is only peripherally related to the subject matter of the present invention.

BenAri does ***not*** address the issue of ***reducing a program*** *while preserving branching time properties*. In addition, BenAri does not disclose or suggest, among other important limitations, computing an abstract program with a reduced number of states and an altered version of said branching time property, f, using said product, as required by independent claims 1 and 14.

Conclusion

The rejections of the cited claims under sections 102 and 103 in view of BenAri are therefore believed to be improper and should be withdrawn. The remaining rejected dependent claims are believed allowable for at least the reasons identified above with respect to the independent claims.

The attention of the Examiner and the Appeal Board to this matter is appreciated.

Respectfully,

Kevin M. Mason

Date: July 28, 2008

Kevin M. Mason
Attorney for Applicant(s)
Reg. No. 36,597
Ryan, Mason & Lewis, LLP
1300 Post Road, Suite 205
Fairfield, CT 06824
(203) 255-6560

APPENDIX

1. A method for reducing a program, M, that preserves at least one branching time property, f, comprising the steps of:

forming a product of said program, M and said branching time property, f, expressed as an automaton, A;

obtaining an abstract domain containing a set of abstract values to generalize possible states of said program and abstract relations that relate said program states to said abstract domain; and

computing an abstract program with a reduced number of states and an altered version of said branching time property, f, using said product.

2. The method of claim 1, further comprising the step of performing an automated program check.

3. The method of claim 2, wherein said automated program check is a model checking step.

4. The method of claim 3, wherein said automated program check is performed for an altered branching time property.

5. The method of claim 1, wherein said computing step further comprises the step of defining a set of states, S', in said abstract program as $S' = \bar{S} \times Q$, where S is a set of states in said program, M, and Q is a set of states of the automaton, A.

6. The method of claim 5, wherein OR states in said set of states, S', are those states where $\delta(q, true)$ has the form $q_1 \vee q_2$ or $\langle a \rangle q_1$, and all other states are AND states, where q are individual states and δ is a transition relation between states.

7. The method of claim 5, wherein an abstract state $(t, \hat{q})$ is in a subset of initial states, I', of the abstract program if there exists $s \in I$ for which $s \xi_{ij} I$, where s is an individual state, I is a subset of initial states, I, of the program, M, and ξ_{ij} is one of said abstract relations.

8. The method of claim 5, wherein for an abstract AND state (t, q), the transition $((t, q); (t', q'))$ is in an abstract transition relation, R', if there exists a concrete state (s, q) and a successor (s', q') that are related to (t, q); (t', q') respectively.

9. The method of claim 5, wherein for an abstract OR state (t, q), the transition $((t, q); (t', q'))$ is in an abstract transition relation, R', only if for every (s, q) which is related to (t, q), there exists a successor (s', q') which is related to (t', q').

10. The method of claim 8, wherein said product ATS $M \times A$ is abstracted by weakening said transition relations at AND states.

11. The method of claim 9, wherein said product ATS $M \times A$ is abstracted by strengthening said transition relations at OR states.

12. The method of claim 8, further comprising the step of obtaining one or more rank functions and employing said one or more rank functions in an abstract transition relation, R'.

13. The method of claim 8, further comprising the step of obtaining one or more choice predicates and employing said one or more rank functions in an abstract transition relation. R'.

14. A system for reducing a program, M, that preserves at least one branching time property, f, comprising:

a memory; and

a processor operatively coupled to said memory, said processor configured to:

form a product of said program, M and said branching time property, f, expressed as an automaton, A;

obtain an abstract domain containing a set of abstract values to generalize possible states of said program and abstract relations that relate said program states to said abstract domain; and

compute an abstract program with a reduced number of states and an altered version of said branching time property, f, using said product.

15. The system of claim 14, wherein said processor is further configured to perform an automated program check.

16. The system of claim 15, wherein said automated program check is a model checking step.

17. The system of claim 16, wherein said automated program check is performed for an altered branching time property.

18. The system of claim 14, wherein said processor is further configured to define a set of states, S', in said abstract program as $S' = \overline{S} \times \underline{Q}$, where S is a set of states in said program, M, and Q is a set of states of the automaton, A.

19. The system of claim 18, wherein OR states in said set of states, S', are those states where $\delta(q, true)$ has the form $q_1 \vee q_2$ or $\langle a \rangle q_1$, and all other states are AND states, where

q are individual states and δ is a transition relation between states.

20. The system of claim 18, wherein an abstract state $(t, \hat{q})$ is in a subset of initial states, I', of the abstract program if there exists $s \in I$ for which $s \xrightarrow{\varepsilon}_{\eta} t$, where s is an individual state, I is a subset of initial states, I, of the program, M, and $\xrightarrow{\varepsilon}_{\eta}$ is one of said abstract relations.

21. The system of claim 18, wherein for an abstract AND state (t, q), the transition ((t, q); (t', q')) is in an abstract transition relation, R', if there exists a concrete state (s, q) and a successor (s', q') that are related to (t, q); (t', q') respectively.

22. The system of claim 18, wherein for an abstract OR state (t, q), the transition ((t, q); (t', q')) is in an abstract transition relation, R', only if for every (s, q) which is related to (t, q), there exists a successor (s', q') which is related to (t', q').

23. The system of claim 21, wherein said product ATS $M \times A$ is abstracted by weakening said transition relations at AND states.

24. The system of claim 22, wherein said product ATS $M \times A$ is abstracted by strengthening said transition relations at OR states.

25. The system of claim 21, further comprising the step of obtaining one or more rank functions and employing said one or more rank functions in an abstract transition relation, R'.

26. The system of claim 21, further comprising the step of obtaining one or more choice predicates and employing said one or more rank functions in an abstract transition relation, R'.

EVIDENCE APPENDIX

There is no evidence submitted pursuant to § 1.130, 1.131, or 1.132 or entered by the Examiner and relied upon by appellant.

RELATED PROCEEDINGS APPENDIX

There are no known decisions rendered by a court or the Board in any proceeding identified pursuant to paragraph (c)(1)(ii) of 37 CFR 41.37.